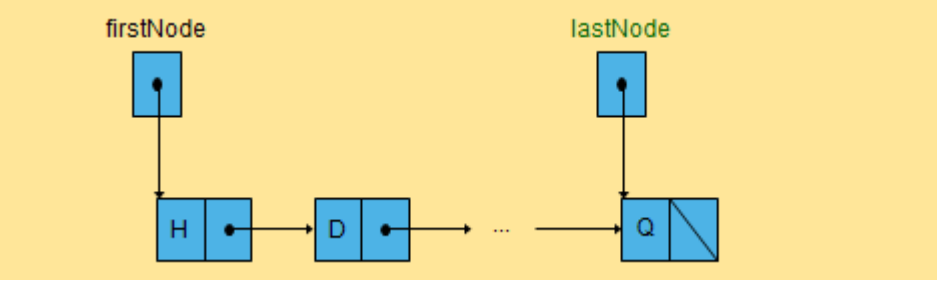
Hogeschool Gent – Faculteit Bedrijf en Organisatie
pb Toegepaste Informatica 2e jaar – Academiejaar 2016-2017
OEFENINGEN OOP3 – Hoofdstuk 21

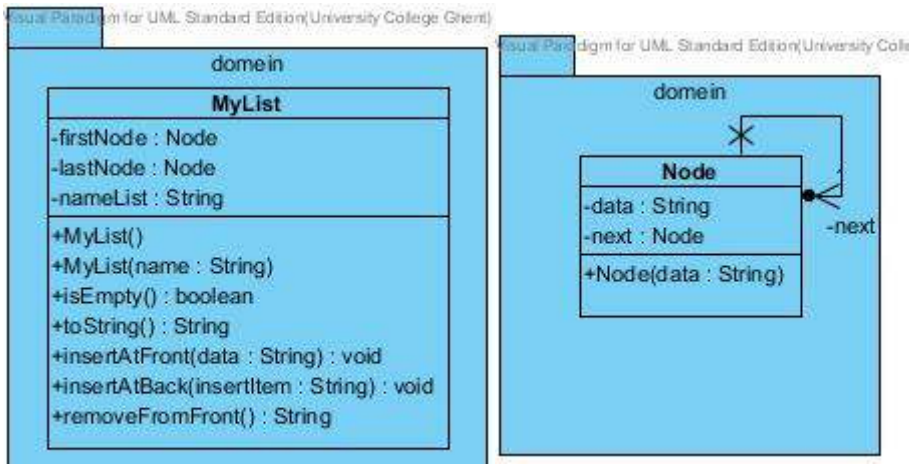
1. Gelinkte lijst
   Maak een enkel gelinkte lijst, een lineaire (sequentiële) collectie van zelf-referentie objecten (= nodes = knopen).

   - Knopen kunnen toegevoegd en verwijderd worden aan het begin, midden of eind van de lijst.
   - Elke knoop bevat een referentie naar de volgende, alleen de laatste knoop heeft niets om naar te verwijzen ➔ de laatste knoop bevat null.
   - Een object bevat de referentie van de eerste knoop van de gelinkte lijst (firstNode). Het object kan ook de referentie van de laatste knoop bevatten (lastNode).



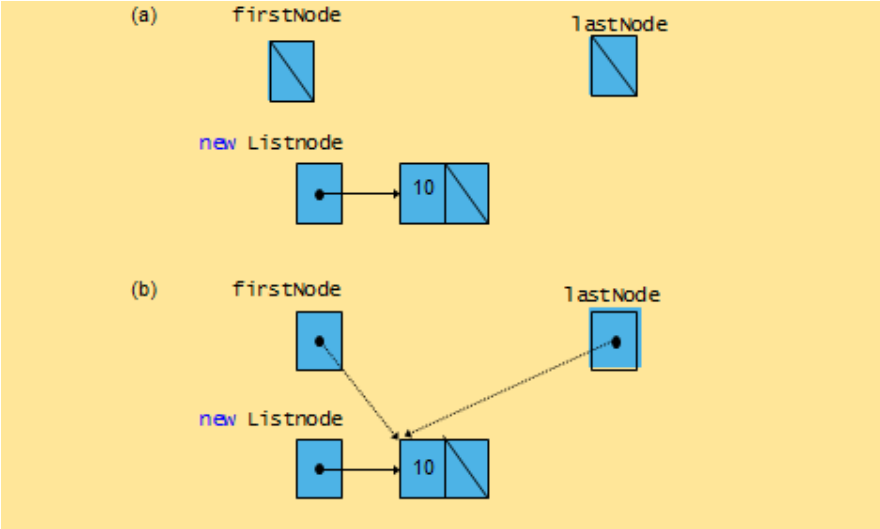
Implementeer de klassen Node en MyList. We nemen String als type voor de data.
MyList:

- Constructor
  In de constructor wordt er een naam aan de gelinkte lijst toegekend.
- Methodes :
- isEmpty : controleren indien de lijst al dan niet leeg is;
- toString : geeft de inhoud van de gelinkte lijst weer als een string;
- insertAtFront : knoop vooraan in de lijst toevoegen;
- insertAtBack : knoop achteraan in de lijst toevoegen;
- removeFromFront : knoop vooraan in de lijst verwijderen;

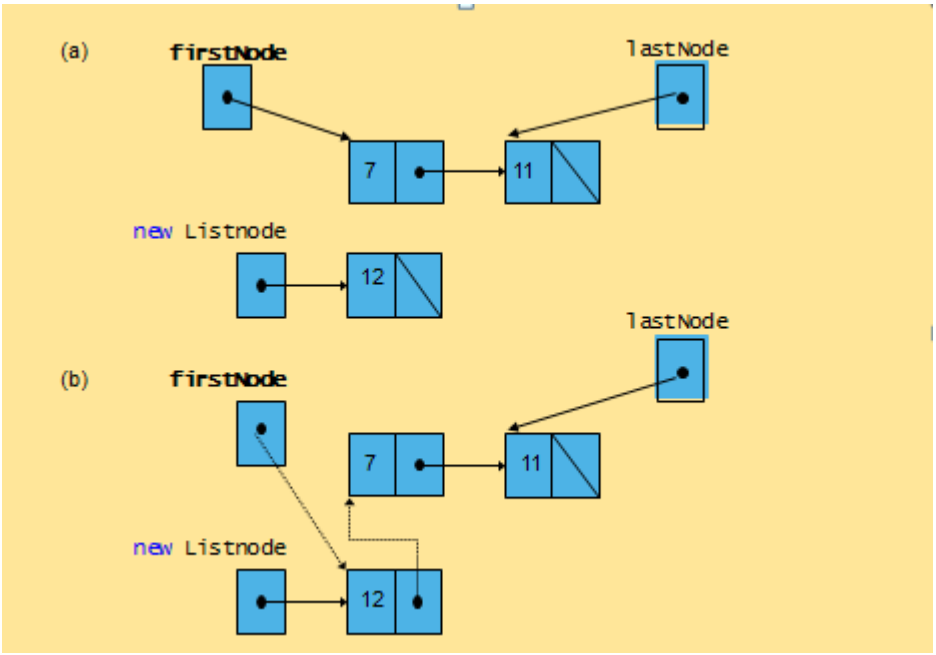


Schematische voorstelling van **insertAtFront:**

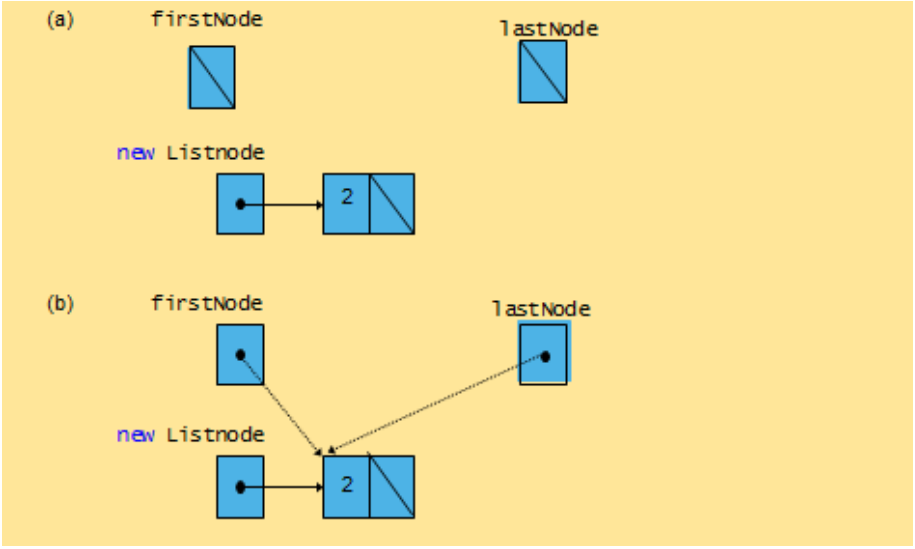
Indien de gelinkte lijst **leeg** is dan zal **firstNode** en **lastNode** naar de nieuwe knoop wijzen.



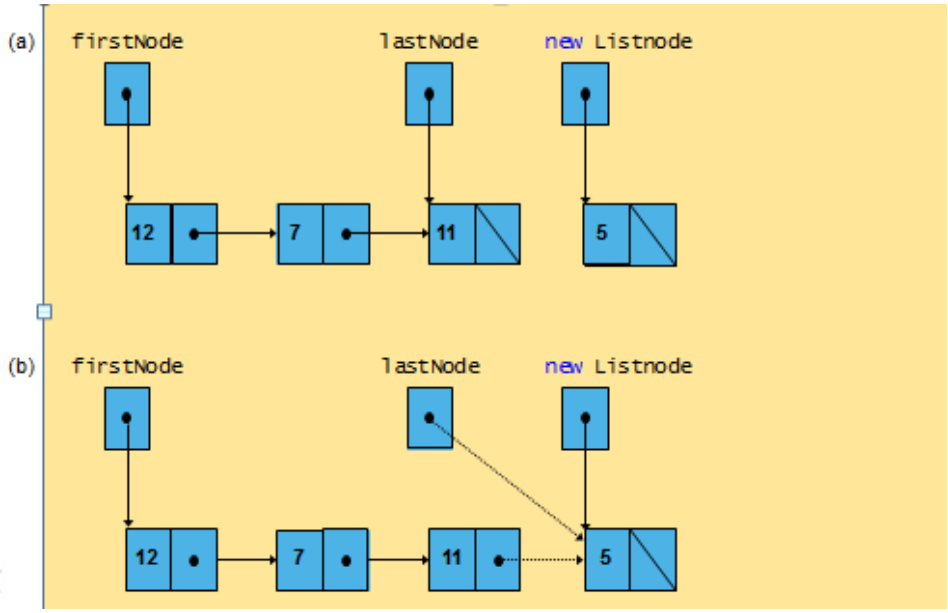
Indien de gelinkte lijst **niet leeg** is dan zal **firstNode** naar de nieuwe knoop wijzen.



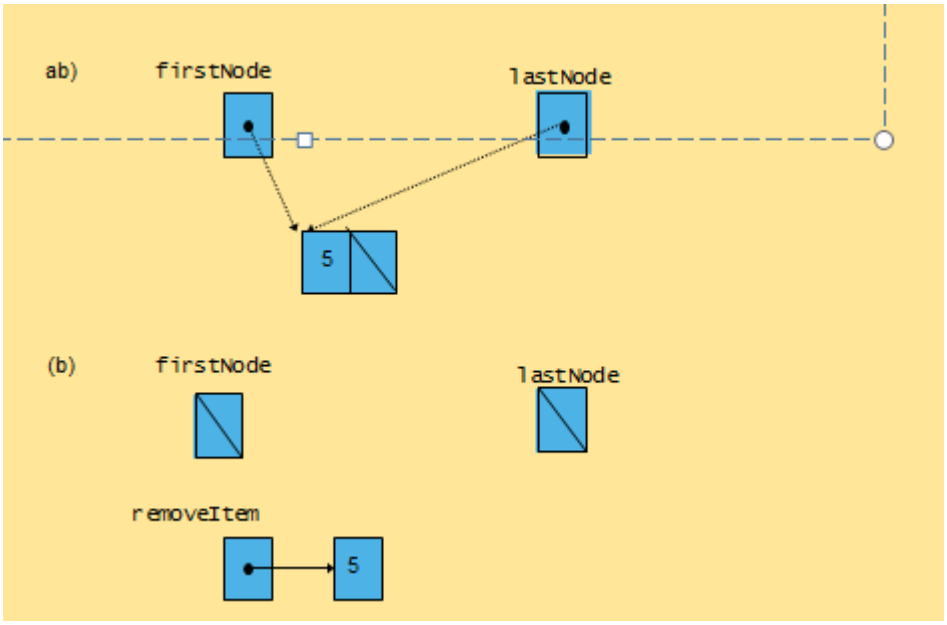
Schematische voorstelling van **insertAtBack:**
Indien de gelinkte lijst **leeg** is dan zal **firstNode** en **lastNode** naar de nieuwe knoop wijzen.



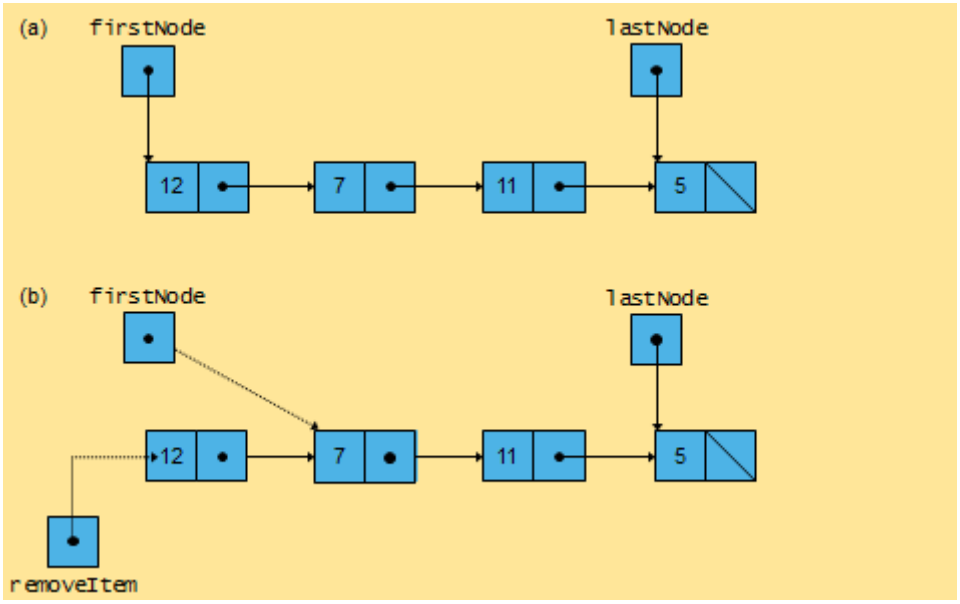
Indien de lijst **niet leeg** is dan zal **lastNode's nextNode** naar de nieuwe knoop wijzen.
Vervolgens zal **lastNode** naar de nieuwe knoop wijzen.



Schematische voorstelling van **removeFromFront** :
Indien de gelinkte lijst één knoop bevat dan zal na deze methode firstNode en lastNode aan null worden toegekend.



Schematische voorstelling van **removeFromFront :**
Indien de lijst **meer dan één knoop** bevat dan zal **firstNode** naar de tweede knoop wijzen.



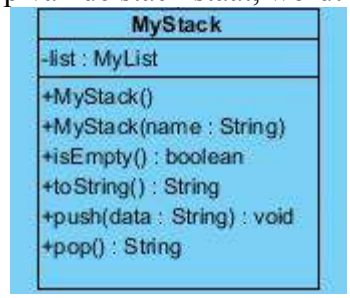
Er wordt een EmptyListException gegooid indien iets verwijderd wordt uit een lege lijst.

2. Stack.

Maak de gegevensstructuur Stack. Een Stack is een gelinkte lijst waarbij de knopen slechts aan één uiteinde (de top van de stack) worden toegevoegd of verwijderd.
Men kan zich een stack voorstellen al een stapel dienbladen die zich op de toonbank van een cafetaria bevindt. Klanten nemen boven van de stapel bladen weg en teruggekomen bladen worden weer boven op de stapel geplaatst. Het blad dat het laatst op de stapel geplaatst is, wordt er het eerst weer afgenomen. Het onderste blad is het eerst in de stapel geplaatst en wordt het laatste weggenomen.

Implementeer

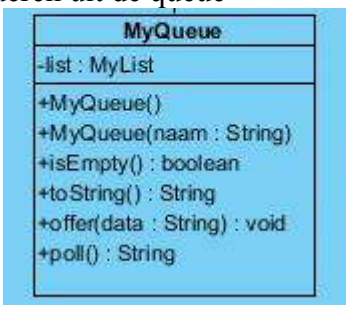
- **push** : nieuwe knoop wordt op de top van de stack geplaatst.
- **pop** : knoop die op top van de stack staat, wordt verwijderd.



3. Queue

Queue (= wachtrij) is ook een beperkte versie van een gelinkte lijst.
Men kan zich een queue voorstellen als een rij bij een loket, waarbij de voorste in de rij als eerste geholpen wordt en nieuwkomers achter aansluiten.
De persoon die het **eerst** in de rij staat, wordt het **eerst** bediend ➔ queue is een F**IFO** gegevensstructuur: first-in, first-out.
Het element van de queue dat vooraan staat in de queue, dat dus als eerste zal worden verwijderd, heet de **kop** van de queue.
Het element dat achteraan staat, dat dus als laatste aan de queue is toegevoegd, heet de **staart** van de queue
We spreken hier niet van push en pop maar van **offer** en **poll**.
Implementeer

- **offer** : element toevoegen aan de queue
- **poll** : element verwijderen uit de queue



4. Toepassing met tekstbestand (gebruik hiervoor de nieuwe klassen uit deze reeks)

Lees alle woorden uit het tekstbestand “Sleutelwoorden” en schrijf ze in omgekeerde volgorde terug naar het bestand “Sleutelwoorden_omgekeerde_volgorde” .

5. Uitbreiding met Iterator
   Voorzie een iterator op de klasse MyList en gebruik die in de implementatie van toString(). Maak een kopie van de klasse MyList met naam MyListIterable en pas de code aan.
   public class **MyListIterable**
   implements Iterable<String>

6. Toepassing met tekstbestand en de klasse MyListIterable.
   Gebruik de klasse MyListIterable om de gegevens uit het tekstbestand "Sleutelwoorden" in op te slaan.
   Doorloop de opgevulde gelinkte lijst en schrijf alle sleutelwoorden die eindigen op een 'e' naar een tekstbestand "Sleutelwoorden_eindigend_op_e" .